cgbind: A Python Module and Web App for Automated Metallocage Construction and Host-Guest Characterization

Tom A. Young, Razvan Gheorghe and Fernanda Duarte

Chemistry Research Laboratory, University of Oxford, Mansfield Road, Oxford OX1 3TA, U.K.

KEYWORDS: Supramolecular metallocages, Host-Guest Complex, Computational Modelling

ABSTRACT: Metallocages offer a diverse and underexplored region of chemical space to search for novel catalysts and substrate hosts. However, the ability to tailor such structures towards applications in binding and catalysis is a challenging task. Here, we present an open-source computational toolkit, *cgbind*, that facilitates the characterization and prediction of functional metallocages. It employs known structural scaffolds as starting points, and computationally efficient approaches for the evaluation of geometric and chemical properties. To illustrate the applicability of *cgbind*, we evaluate the likelihood of 102 substrates to bind within M_2L_4 and M_4L_6 cages and achieve accuracy comparable or better than semi-empirical electronic structure methods. The *cgbind* code presented here is freely available at *github.com/duartegroup/cgbind* and also via a web-based graphical user interface at *cgbind.chem.ox.ac.uk*. The protocol described here paves the way for high-throughput virtual screening of potential supramolecular structures, accelerating the search for new hosts and catalysts.

1. Introduction

Metal-mediated self-assembly has emerged as a promising approach to develop complex biomimetic systems from simple building blocks. The synthetic modularity and tunability, achievable by modifying the metal center or organic linkers, has led to the discovery of an immense diversity of supramolecular metallocages with applications in molecular separation[1,2] medical imaging,[3,4] drug-delivery[5–7] and catalysis.[8] Moreover, the dynamic nature of metal–ligand bonds, intermediate in strength between covalent bonds and weak non-covalent interactions (NCI), facilitates error correction and reversibility.[9,10] The latter being more challenging to achieve with covalent organic cages.[11]

Despite these attractive attributes, the development of new metallocages and their use in real-life applications has been slow, and in many cases driven by trial-and-error rather than rational design. To date, most reported binding studies have been limited to a relatively small set of guests and host combinations. This is primarily due to current synthetic techniques being time-consuming and expensive; while also limited in environmental sustainability.[12] To fully exploit their potential, the development of efficient methods to identify structures with desirable binding or catalytic properties, without having to synthesize all possible structures, is necessary.

Computational screening offers a path to overcome these limitations and accelerate the discovery process.[13] While reaching chemical accuracy (< 1 kcal mol^{-1} error), generally required for quantitative comparison to experiments remains a challenge,[14] current computational and cheminformatic approaches already allow a fast sift of the desired space. For example, molecular descriptors[15–17] and inverse design methods[18] can be used to select (with reasonable accuracy) promising regions of the chemical space for further computational and eventually experimental study.[19] Similarly, efficient electronic structure methods, in particular density functional theory (DFT), have enabled the calculation of structure and properties for hundreds of compounds in a single study.[20,21] While these protocols are accessible for small organic molecules and periodic structures, extension of similar approaches in supramolecular assemblies has been much less explored.[22] We have recently shown that an efficient DFT-based approach can reach a ~2 kcal mol^{-1} level of accuracy in quantifying binding affinities for M_2L_4 metallocage assemblies.[23] However, with a sufficiently fast computational approach, the required time to set up and analyze a new set of metallocages rapidly increases as the size of the building block library grows. Therefore, methods to automate these steps are becoming increasingly valuable.

In the field of metal-organic frameworks (MOFs), several computational screening tools have been developed in recent years.[24,25] They include open-source tools to screen existing MOF libraries (*Pymatgen*[26] and *Zeo++*[27]), deconstruct known structures in their building blocks,[28] and evaluate their catalytic

proficiency for specific reaction (*PyMOFScreen*[29]). In the space of metallocages, however, no large already synthesized libraries exist; thus, tools that both generate and rank structures for desired properties are required. For porous organic cages, *in silico* toolkits have been developed for constructing and predicting their adsorption properties via a combination of building blocks and high-throughput screening (e.g. *stk*, *pywindow, Eigencages*).[17,30–34] Current toolkits to generate structures are, however, not directly transferable to metal-containing systems. Therefore, a different methodology is required to further enable high-throughput analysis of supramolecular metallocages.

Here, we describe *cgbind*, an open-source Python API and its web-based graphical user interface (GUI). We first describe the methodology and its implementation and proceed to demonstrate the ability of *cgbind* to explore the space of metallocages by combinatorial cage generation from a building block library. Furthermore, we show how *cgbind* can be used to generate accurate geometries and obtain estimated binding affinities.

2. Results and Discussion

2.1 Software Outline

cgbind is written in object-oriented Python and employs scientific programming libraries, including Numpy, Scipy and NetworkX.[35–37] The code is built around four key objects (**Linker**, **Cage**, **Substrate** and **CageSubstrateComplex**), the relation and inheritance between which, their attributes, and key methods are outlined in **Figure 1**.

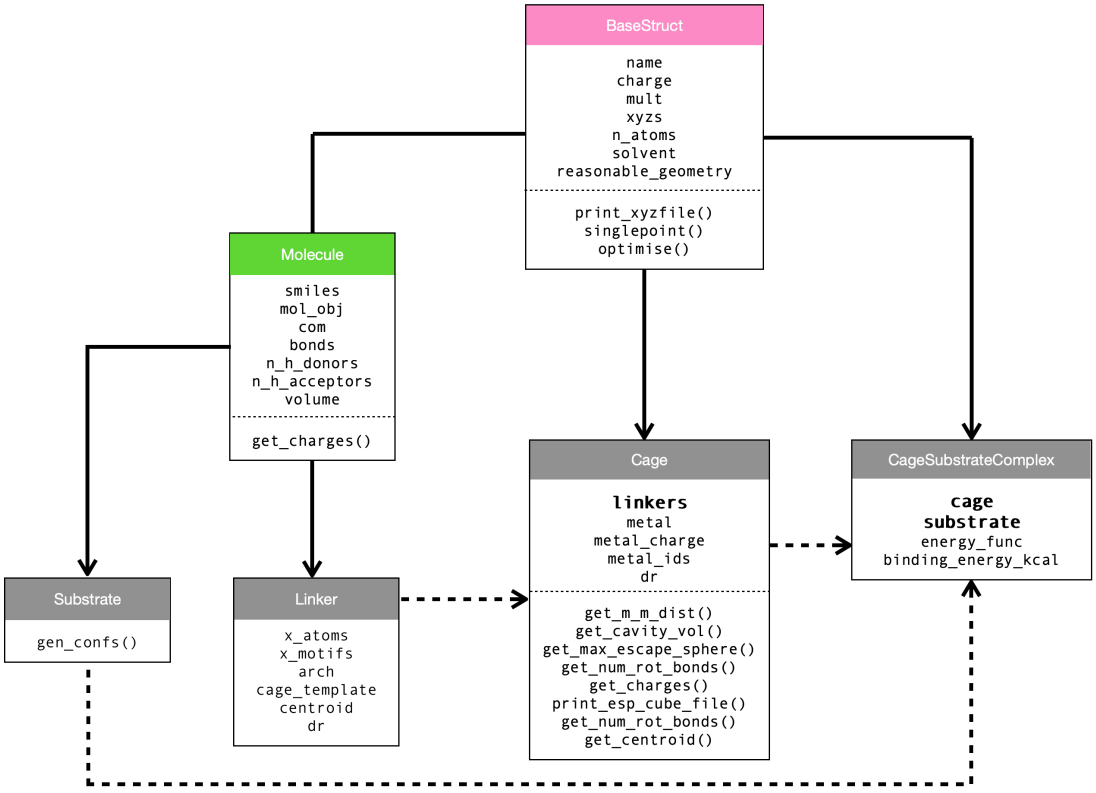


Figure 1. Class structure of *cgbind*. A solid line from A to B indicates that B inherits from A and a dashed line B is built from A.

A homoleptic **Cage** (M_xL_y with x metals and y linkers) is initialized from a single **Linker** object, while heteroleptic (e.g. $M_2L_2L'_2$) variants from a set of different **Linkers**. All **Molecule** objects and their subclasses are designed to be initialized from SMILES strings,[38] with their corresponding 3D structures generated using conformer generation algorithms implemented in RDKit.[39,40] While this affords rapid access to 3D structures, conformers are occasionally not adequately generated (*vide infra*). **Molecules** can also be initialized from a variety of molecular file formats (.xyz, .mol, .mol2) using the *input_output* module. Upon **Cage** construction a number of analysis tools are available as class methods. The analysis

of cage-substrate complexes (**CageSubstrateComplex** objects) is also possible. They are built from a **Cage** and one **Substrate**, where a substrate may contain more than one molecule, thus multiple guests are supported. Furthermore, a **Substrate** can either be a minimum or a transition state (/analogue), providing a route to predicting catalytic activity.

Binding affinities between a substrate (S) and cage (C) [$\Delta E_{BA} = E_{C\text{-}S} - (E_C + E_S)$] are calculated in the first instance using a simple and general forcefields developed here (*vide infra*). More accurate estimates can be obtained from electronic structure codes, by invoking *singlepoint*() and *optimise*() functions, which deliver energies (**Molecule**.*energy*) and optimized structures (**Molecule**.*xyzs*). These functions are currently handled through a **Calculation** object in our Python API *autodE*, which facilitates the automated generation of reaction profiles (github.com/duartegroup/autodE). This enables *cgbind* to utilize electronic structure packages with currently implemented wrappers (inc. GFN2-XTB,[41] MOPAC,[42] ORCA[43] and NWChem[44]).

The structure of *cgbind* benefits from modularity, allowing analysis functions to be implemented by simply adding methods to the **Cage** or **CageSubstrateComplex** objects. The fragment approach to generating a **CageSubstrateComplex** allows for all the properties of the cage, and constituent linkers to be contained within a single object. **Linker**, **Cage** and **Substrate** objects may therefore be reloaded from a saved (e.g. Pickled) **CageSubstrateComplex** object, allowing for data mining within a saved library. Independent calculations are parallelized up to the user-defined number of cores set in the **Config** object. The software is distributed under the MIT license and is provided with unit tests and comprehensive online documentation including examples.

2.2 Overview of *cgbind*

2.2.1 Metallocage Construction

Broadly, metallocage structures are generated from a SMILES or 3D representation of the linkers, which are templated onto known or user defined metallocage templates. The methodology employed is not template specific, such that extension to architectures other than those provided with the code (M_2L_4, M_4L_6, M_6L_8, $M_{12}L_{24}$) is straightforward. **Figure 2** provides an overview of the workflow behind *cgbind*, which can be broken down into six key steps:

Step I. Generate *n* linker (L) conformers from a SMILES string, a common way of representing molecules, which can be obtained from online databases (e.g. PubChem) or chemical software (e.g. ChemDraw®).

Step II. Locate coordinatively unsaturated heteroatoms (*X*-atoms) in L.

Step III. Locate possible *X*-motifs, containing one or more *X*-atoms and their nearest neighbors in L.

Step IV. Discard any *X*-motifs with a different number of atoms to the template *X*-motifs.

Step V. Minimize the fitting cost with respect to linker & template parameters. The fitting cost is defined as the sum squared distance (SSD) between the X-motif atoms in the template and L.

Step VI. Fit linkers to the expanded/contracted template, controlled by a distance Δ*r*, while keeping steric clashes below a threshold.

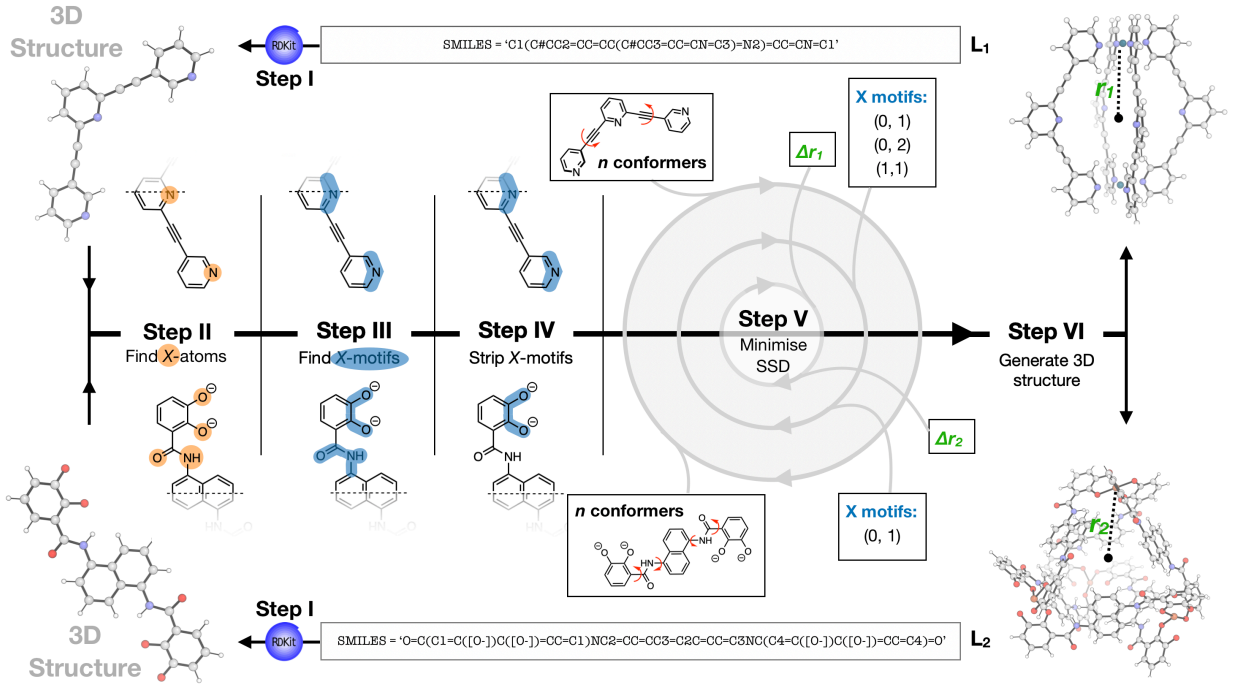


Figure 2. Methodology used to construct metallocages in *cgbind*. Centroid–metal distances are denoted as *r* and shifts Δ*r*. *X*-motifs are indexed from zero to the total number in the linker.

These steps are discussed in detail below for two architectures, M_2L_4 and M_4L_6, already saved in the template library. Note that, depending on the input provided by the user, not all steps must be executed. For example, if a **Linker** is initialized from a 3D geometry then the Step I can be skipped. Moreover, if the *X*-motifs are specified by the user then steps III and IV may be skipped, as well as minimization with respect to *X*-motif sets in Step V.

For a bis-pyridine linker found experimentally to form a M_2L_4 assembly (**L**$_1$, top **Figure 2**),[45] the first step involves generation of 3D conformers using the Experimental-Torsion Distance Geometry (ETDG) algorithm[40] implemented in RDKit[39] (Step I). For this linker, requesting 300 conformers and using a 0.3 Å

root mean square displacement (RMSD) threshold to compare them, renders 50 conformers. This provided a good balance between speed and high probability of locating a suitable conformer. Subsequently, *X*-atoms are identified as the heteroatoms with at least one 'lone pair' capable of metal donation (here the nitrogens, Step II). Combining *X*-atoms and their nearest neighbors afford three *X*-motifs containing CNC atoms (Step III). In the M_2L_4 template, the *X*-motifs also contain three atoms; therefore, all *X*-motifs are kept (Step IV). Minimization of the fitting cost is then performed exhaustively with respect to all conformers and *X*-motif sets, and minimization with respect to expansion and contraction of the template is achieved using the Broyden–Fletcher–Goldfarb–Shanno (BFGS) optimization algorithm implemented in *SciPy*[36] (Step V). The template is expanded/contracted by shifting the *X*-motifs closest to metal *i* by the M_i–centroid vector a distance Δr. For $\mathbf{L_1}$, there are three *X*-motifs (top: 0, middle: 1 and bottom: 2, with 0 and 2 being equivalent), while there are only two *X*-motifs in the template linker; therefore, the three *X*-motifs in $\mathbf{L_1}$ can be fitted in three different ways, i.e. (0, 1), (0, 2) and (1, 2). *X*-motif atoms are fit to the template using an implementation of the Kabsch algorithm.[46] Step V correctly identifies the linker with the most coplanar pyridyls as the optimum conformer, the two terminal CNC atom sets (0, 2) as the optimum selection of *X*-motifs and a small shift required in the template ($\Delta r_1 = -0.0037$ Å). Upon sequentially fitting the optimum *X*-motifs in the linker to the adjusted template and adding two metal ions, a metallocage geometry is generated (Step VI).

In an M_4L_6 structure formed from identical catechol-derived linkers and Fe^{2+} ($\mathbf{L_2}$, **Figure 2**),[47] steps I/II proceed as for $\mathbf{L_1}$, while identification of *X*-motifs is slightly more complex. *X*-atoms are joined by their nearest neighbors to generate *X*-motifs for C-O^-, C=O and CN(H)C and those separated by a one bond joined generating $^-$OCCO$^-$ and O=CN(H)C motifs to give a total of 12 in $\mathbf{L_2}$ (Step III). The template contains *X*-motifs with four atoms, thus all but the $^-$OCCO$^-$ *X*-motifs remain upon discarding those with greater and fewer than four atoms (Step IV). Steps V and VI proceed as for $\mathbf{L_1}$ but requires no enumeration over *X*-motif sets as both $\mathbf{L_2}$ and the template contain two *X*-motifs.

This strategy is amenable to heteroleptic cage generation; different linkers are generated within the same architecture and fitted in the order defined by the template. The template shift distance Δr is taken as an average over all linkers, as to accommodate linkers with (slightly) differing lengths. For an experimentally accessible *cis*-$M_2L_2L_2$' metallocage[48] the process is shown in **Figure 3**. We note that the *trans* isomer may be generated by simply reordering the linkers in the **Cage** initialization (i.e. linkers=[linker1, linker2, linker1, linker2] in **Figure 3**).

```python
from cgbind import Linker, Cage, XTB, Config

# Use 8 cores throughout the construction and optimisation
Config.n_cores = 8

# Initialise a cage from two linkers to generate a M2L2L'2 cage
linker1 = Linker(smiles='O=C1C2=C(C=CC(C#CC3=CC=C(C)N=C3)=C2)NC4=C1C=C(C#CC5=CN=C(C)C=C5)C=C4',
                 name='L_A', arch_name='m2l4')
linker2 = Linker(smiles='O=C1C2=C(C=CC(C#CC3=CC=CN=C3C)=C2)NC4=C1C=C(C#CC5=C(C)N=CC=C5)C=C4',
                 name='L_B', arch_name='m2l4')

# Construct a metallocage, optimise with XTB and print the geometry as a .xyz file
cage = Cage(linkers=[linker1, linker1, linker2, linker2], metal='Pd', metal_charge=2)
cage.print_xyzfile()
cage.optimise(method=XTB)
```

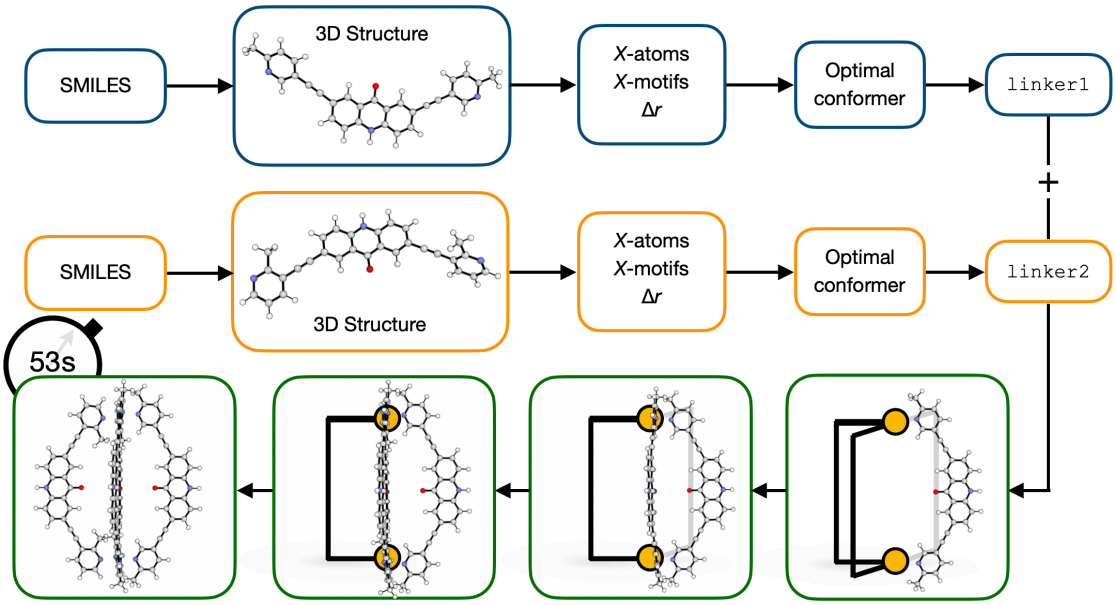


Figure 3. Formation of a heteroleptic cage within *cgbind*. Total execution time includes XTB optimization.

For a linker with *n* conformers, *j X-motifs* in the template and *i* metallocage *X*-motifs, the minimization is conducted in a $\sim n \times {}^{i}C_{j}$ dimensional space. However, despite this large space, generally it is the conformer generation in RDKit, rather than minimization which dominates the execution time. Construction of cages with non-planar and or conformationally flexible linkers is facilitated by checking on the ordering of *X*-motifs and linker conformer as to achieve a threshold in the linker–partial cage repulsion during the build algorithm.

2.2.5 Adding New Template Architectures

Extensibility to new architectures is an essential feature of the code. The inclusion of new architectures by the user is achieved as follows: A 3D structure file (.mol2) is used for initialization of a **Template**. This structure can be downloaded from the CSD, or provided by the user (from non-deposited crystal structures or e.g. Spartan[49] generated constructs).

(1) Utilizing NetworkX a molecular graph of the system is generated;
(2) A metallocage is identified as the molecular component with the largest number of metal atoms. All other molecules (ions, substrates) are stripped out; this step is required if the template is initialized from a crystal structure;
(3) The donor (*X*-) atoms bonded to the metal atoms are located;
(4) Linkers are identified by deleting metal atoms and determining its connected components using NetworkX;
(5) Within the linkers, and from the previously located *X*-atoms, *X*-motifs are located as the maximally connected set of *X*-atoms and their nearest neighbors;
(6) The shift vector for each *X*-motif is calculated to enable symmetric expansion of the template controlled by Δr as above.

Overall, this workflow facilitates the addition of novel architectures into the code with minimal human intervention (i.e. a single command and a canonical template) and without structures needed to be hardcoded.

2.2.2 Analysis Tools

Upon generation of a metallocage structure, *cgbind* provides several geometry-based parameters that can be used to analyze and select metallocages for further binding and/or catalysis studies (**Figure 4a**). These include:

a. *Number of rotatable bonds;* which may be used as a proxy for cage flexibility (by default, the M-L bonds are not included).
b. *Total number of H-bond;* which can be used as a proxy for hydrophilicity. They include H-bond donors (non-coordinatively saturated heteroatoms) and acceptors (hydrogens bonded to a heteroatom).
c. *Cage size metrics*. They include:

(i) Average M–M distance,
(ii) (Maximum enclosed sphere, defined as the largest sphere centered on the cage centroid that does not contain any atomic (van der Waals) volume. The cage centroid is defined as the mean position of the metal atom centers;
(iii) Maximum escape sphere, defined the largest sphere that may be removed from the cavity. This volume is determined by maximizing the sphere size at a distance *r* from the cage centroid and taking the minimum sphere size along the path to the outside of the cage.

The latter process can be performed with a basin-hopping algorithm from the *SciPy* library, which provides a more accurate estimate of the extrema but is computationally demanding.[50]. If the cage is

symmetric, initializing with a random vector generally affords the largest escape sphere. However, for asymmetric cages the use of a basin-hopping algorithm is necessary. This methodology is similar to that employed in *pywindow*[34] and may be used as a proxy for window size, and so substrate diffusion rates into the interior of the cage.

d. Electrostatic potentials (ESP), which can be used to identify favorable guest-host complexes by maximizing electrostatic interactions between them. By interfacing with the open-source tight binding DFT code XTB,[41,51] ESP are also available for metallocages in seconds (**Figure 4b**) in the standard Gaussian cube format, which may be visualized in PyMOL or other graphical software. This is achieved by performing a single point energy evaluation on the structure, retrieving the partial atomic charges and constructing the ESP on a uniform grid as $\mathrm{ESP}(\boldsymbol{r}) = \Sigma_i\, q_i/|\boldsymbol{r} - \boldsymbol{r}_i|$ in atomic units, where i enumerates over atoms. Note that this approximation does not result in any loss in qualitative accuracy compared to the true integral over the electron density (**Figure S1**).

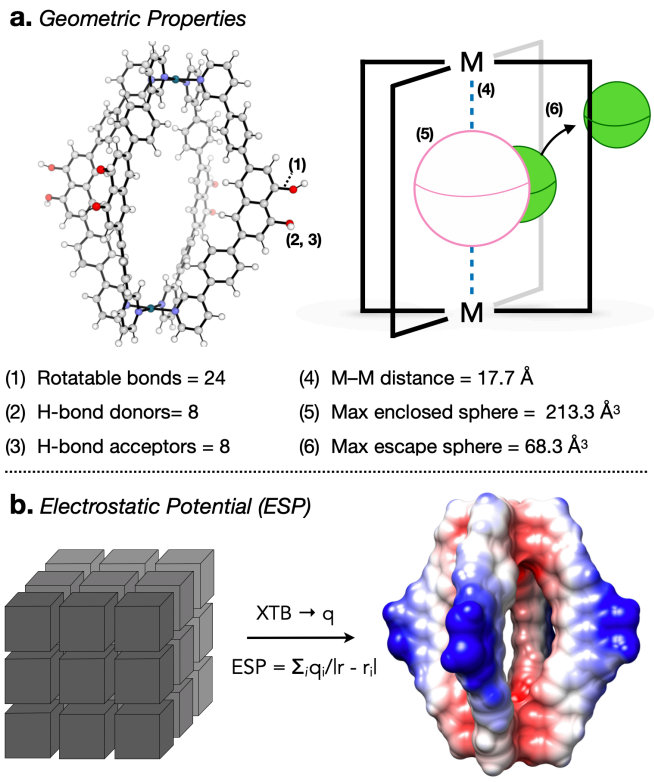


Figure 4. Calculated metallocage properties for an archetype M_2L_4 cage. (a) Geometrical properties: **(1)** rotatable bonds, **(2)** H-bond donors, **(3)** metal-metal distance, **(4)** maximum enclosed sphere and **(5)** maximum escape sphere. (b) Electrostatic potential (ESP) calculated from XTB partial atomic charges enumerated over a grid where blue/red corresponds to low/high values of the ESP.

2.2.3 Substrate Encapsulation and Binding affinity prediction

Once a metallocage is constructed, a substrate can be added to generate a cage-substrate complex. This is achieved by adding the substrate such that its center of mass (COM) is located at centroid of the cage. The system is then energy minimized with respect to the substrate rotation and conformer, i.e. the global minimum on the rigid-body PES for each conformer is located (**Figure 5**). To obtain the global minimum a simple (random + BFGS) optimization algorithm implemented in RDKit is employed,[36] requiring ~$n \times m \times o$ energy evaluations, where n is the number of conformers, m the number of initial rotations and o the number of BFGS optimization steps (~20). We have found that m = 50 affords the global minimum with probability ~0.8 for the systems tested (**Figure S3**). The number of conformers to screen is of course substrate dependent and is, as such, user specified.

Considering the rather large number of energy evaluations that would be required to evaluate binding of each conformer, electronic structure theory is not viable. To speed up this process we instead resort to empirical forcefields (FF). While a number of general FFs are available, their use is labor-intensive, requiring the tabulation of parameters and/or the use of specific molecular simulation packages. Furthermore, inherent approximations in their development limit their accuracy. To circumvent and simplify this problem, and assuming that substrates-cage interactions are dominated by sterics, we employed the following empirical energy function:

$$E_{int}(r+k) = \Sigma_{i,j \in \text{pairs}}\ c\ \exp(a - |\boldsymbol{r}_i - \boldsymbol{r}_j|/b) + k \times S_{atoms} \qquad \textit{(eq. 1)}$$

Where the first half is a pairwise repulsive term, similar to the one found in a Buckingham potential, with c = 1 and with units of energy. Using noble gas dimers as as a model for closed shell repulsion, we found that a and b are roughly linearly dependent on the sum of van der Waals radii (**Figure S2**). The second term is a constant attractive contribution (k) based on the number of substrate atoms (S_{atoms}). k = 0.4 kcal mol^{-1} atom^{-1} was derived as the optimum classifier for the set of binary binding data outlined in **Table S2** (**Figure S4**). This approximation is considerable, as the attractive terms are dependent on the nature of non-covalent interactions (NCI) between the substrate and the cage (dispersion, H-bonding, etc.). Therefore, we only suggest using this FF to generate cage-substrate complexes where there the attractive term is expected to be small. We refer to this method as 'repulsive' ($r+k$) and the binding energies arising from it as $E_{int}(r+k)$ (energy_method='repulsion').

We have also considered a more physical FF, which utilizes repulsive and electrostatic terms. The latter is calculated using either Gasteiger atomic partial charges from RDKit, which do not account for polarization of the atomic centers due to the surrounding environment (*fe*, energy_method='electrostatic_fast'), or

alternatively, charges derived from an XTB single point calculation which account for polarization effects (*e*, energy_method='electrostatic'). These methods more accurately account for NCI between the substrate and the cage. For example, within $[Pd^{II}_2L_4]^{4+}$ metallocages, both schemes delivers the expected binding modes, which involves hydrogen bond interactions between the quinone oxygen and the α-C-H groups at the top/bottom of the cage (**Figure 5**).

Quantifying the binding affinity of a given cage for a substrate is key in determining their applicability in real-world applications. However, its quantification is challenging as key sources of errors in the calculations of free energies include inadequate quantification of entropy, sampling and errors in the methodology used. We recently showed that potential energy difference is sufficient to obtain quantitative agreement with experimental binding free energies for M_2L_4 architectures.[23] The calculation of thermal contributions to the enthalpy is more computationally demanding, requiring calculation of the Hessian matrix (i.e. requiring frequency calculations), while estimation of binding entropies is difficult.[52]
In *cgbind*, once the cage-substrate complex has been generated, an approximate interaction energy (E_{int}) based on a simple FF above, is available immediately, as it is minimized in the construction. This gives a rough estimate of the binding affinity and the likelihood of a cage to bind/not-bind a given substrates (*vide infra*). A more accurate estimate can be obtained from an electronic structure theory, calculating either potential or Gibbs free energies.

```python
from cgbind import Cage, Linker, Substrate, CageSubstrateComplex, XTB

# Generate a cage and optimise with XTB
linker = Linker(name='L', smiles='C1(C#CC2=CC=CC(C#CC3=CC=CN=C3)=C2)=CC=CN=C1', arch_name='m2l4')
cage = Cage(linker, metal_charge=2, metal='Pd')
cage.optimise(method=XTB)

# Generate a substrate and optimise with XTB
substrate = Substrate(name='benzoquinone', smiles='O=C1C=CC(C=C1)=O')
substrate.optimise(method=XTB)

# Build the substrate@cage complex
cs_complex = CageSubstrateComplex(cage, substrate, energy_method='electrostatic_fast')
cs_complex.print_xyzfile()
```

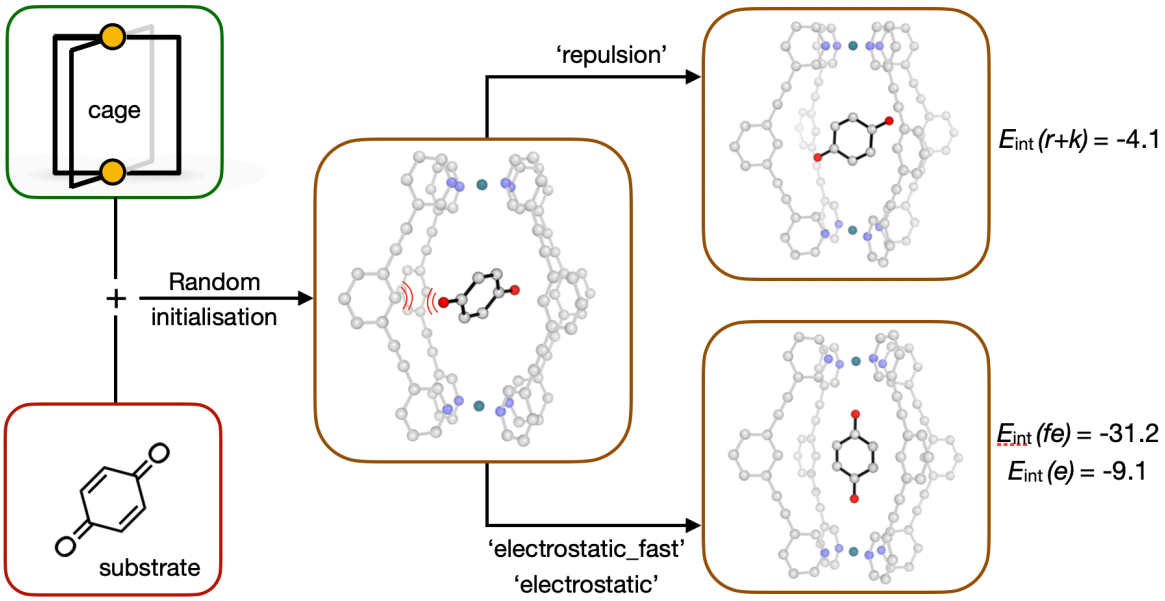


Figure 5. Metallocage-substrate complex construction and Interaction energy calculation for benzoquinone inside a $[Pd^{II}_2L_4]^{4+}$ cage using the simple repulsive (*r*+*k*) and electrostatic forcefields (*fe* and *e*). Interaction energy predictions are quoted in kcal mol^{-1} and do not include thermal or entropic contributions. Hydrogen atoms omitted for clarity.

2.2.2 Web app GUI

Alongside the Python interface we have also developed a web application as a GUI to the module. This approach allows a user to interact with key components of the code within a web browser, thereby reducing the activation barrier to its utilization. Furthermore, by developing a web app the user does not need to download or install any packages/dependencies, which in many cases can be limited to specific platforms (Windows/Mac/Linux). The app is currently available at cgbind.chem.ox.ac.uk and provides the ability to: a) generate and visualize a selection of metallocages, b) perform analysis of metallocage size, c) compute and visualize electrostatic potential maps, and d) perform substrate binding calculations. SMILES strings of linkers can be imported from chemical software packages, or generated in the *cgbind* GUI using the 2D drawing tool Kekule.js.[53] Interactive molecular visualization of the generated cage is enabled using the open-source 3Dmol.js package, allowing useful actions such as rotating and zooming.[54] The whole app

distributed in a microservice architecture [front-end (Flask), web-server (Gunicorn/nginx), message-broker (Redis), database (PostgreSQL) and task queue (Celery)] using Docker.

2.3 Examples

2.3.1 Geometric Accuracy

To explore the ability of *cgbind* to generate cages with varied topologies and linker functionalities we generated nine metallocages (**a**–**i**, **Figure 6**) and computed the root mean squared displacement (RMSD) of the heavy atoms to known crystal structures. With the exception of $\mathbf{L_h}$ (*vide infra*) all the metallocages were successfully generated from just the linker SMILES string all with RMSD < 1.5 Å. Geometry optimization of the structures at the tight-binding DFT level did, in some instances, improve RMSDs while for others (**d**, **g**, **i**) a larger deviation was obtained with the structure falling into a different local minimum. Similarly, optimization at the DFT level of theory (PBE-D3BJ/def2-SVP) provides no improvement in RMSD. This observation highlights the importance of accounting for the conformational flexibility of the linker and cage when constructing and analysing metallocages and their properties. We believe that this conformational variability provides an interesting avenue to explore substrate-dependent metallocage structure and function as well as a variety of allosteric effects.

To generate a cage, the algorithm requires a conformer with donor atoms in the correct orientation. While conformational generation algorithms available in RDKit (ETKDG, ETDG) are generally adequate, for $\mathbf{L_h}$ (**Figure 6**), they do not afford a conformer with the adequate torsions to generate a reasonable metallocage of M_6L_8 geometry, even with 10^4 conformers requested. This may be due to the RMSD threshold employed (>0.3 Å RMSD), which may remove relevant conformers or to the lack of a much more extensive sampling of the conformational space. Similarly, the open-source conformer generation implemented in OpenBabel (CONFAB),[55] which performs well at generating structures close to that found in the crystal structure, does not afford the correct conformer from the RDKit initialized 3D structure.[56] We note that the required conformer of $\mathbf{L_h}$ is 0.6 kcal mol^{-1} more stable than the first RDKit generated analogue (**Figure S5**) and therefore should be generated in the conformer ensemble. Cage **h** was therefore constructed from $\mathbf{L_h}$ built manually in the correct conformation and a **Linker** object initialized from an xyz file. Nevertheless, we consider our methodology to be robust in generating metallocages for varying architectures and multiple possible donor atoms. We also note that *cgbind* provides reasonable structures in seconds to minutes rather than in hours (XTB) or weeks (DFT) for the largest metallocages shown in **Figure 6**.

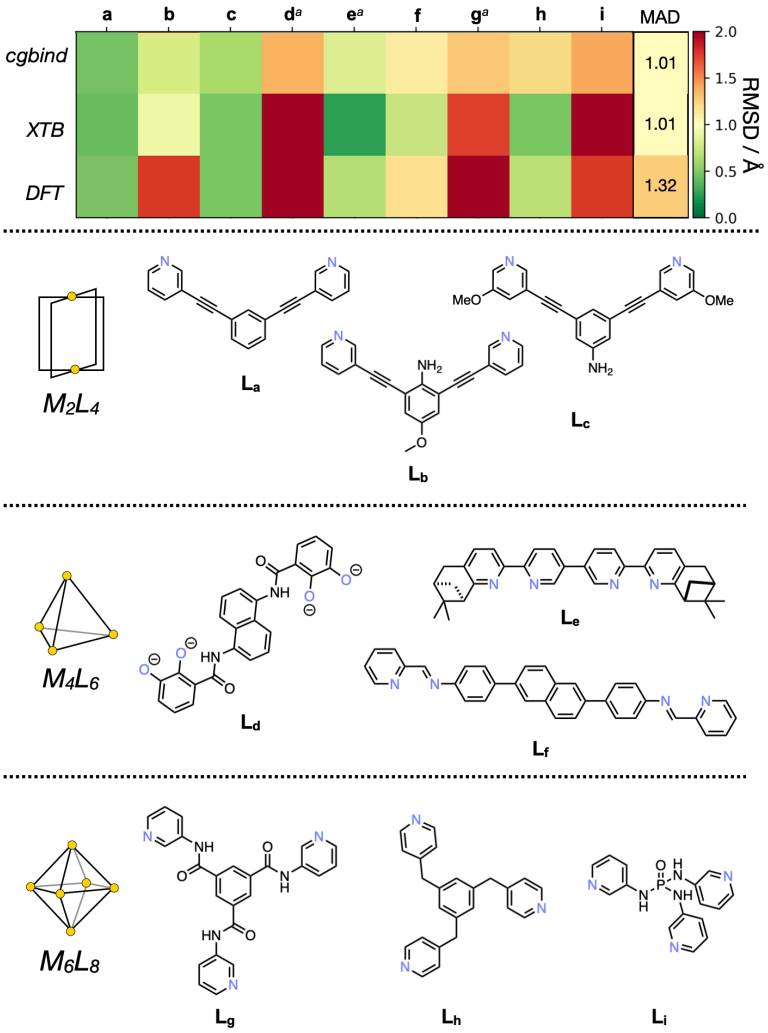


Figure 6. Root mean squared deviation (RMSD, Å) and their mean absolute deviations (MAD) of *cgbind*-generated metallocage structures relative to available crystal structures in the CSD as a function of increasing computational expense, only heavy atoms were considered for the RMSD calculation. Counterions/solvent molecules in the crystal structures were removed for the analysis. Data in **Table S1**. *a*. DFT geometries converged to a criteria only on forces (RMS(grad) < 5 mHa a_0^{-1}).

2.3.2 Binding Accuracy

To assess the accuracy of the classification of substrates into binding and non-binding, we studied the binding of 102 host-guest complexes reported in literature, including M_2L_4 and M_4L_6 host architectures and a wide range of polar and apolar substrates and solvents (dichloromethane, acetonitrile and water; **Figure 6, Table S2**). They include:

1. 12 quinone-type guests which host–guest properties with two different $Pd^{II}_2L_4$ cages (24 host-guest complexes in total), in dichloromethane and acetonitrile, were studied by Lusby et al.[57] Depending on the nature of the quinone guest, binding constants of up to 10^8 M^{-1} have been observed for these systems. Many of them have also been found to be useful substrates in Diels-Alder reactions and cofactors in radical-cation cycloadditions.[45,58]
2. Six different guests, including polycyclic aromatic hydrocarbons and steroids, with a series of nine aromatic-panelled $Fe^{II}_4L_6$ tetrahedral cages studied by Nitschke et al. (54 hots-guest complexes in total).[59] In this cage series, the size and arrangement of the aromatic panels were found to dictate guest binding propensity.
3. 24 neutral and charged guests bound to $Fe^{II}_4L_6$ tetrahedral cages studied by Nitschke et al.,[60] of which 21 were experimentally found to bind with binding constants ranging from 3 up to 10^4 M^{-1}. The host-guest properties of these systems in aqueous solution were found to strongly dependent on the hydrophobicity of the guest.

Several practical obstacles make it difficult to quantitatively determine and compare binding strength across different studies, including guest solubility and the use of different techniques. Therefore, as a first approximation, we consider the experimental data and our prediction as binary classifiers, i.e. either the guest is observed to bind or not within the cage. The binding affinity is considered to be accurate if $E_{int} < 0$ and experimentally the substrate binds, or alternatively $E_{int} > 0$ where experimental binding is not observed.

Cage substrate complexes were thus initialized using a single conformer in most cases, with the exception of cholesterol where 50 conformers were considered. Interestingly, our simple repulsive (r+k) and both fast and normal electrostatic forcefields (*fe* and *e*, respectively) perform as well as calculating the interaction energy at the tight binding DFT level of theory (**Figure 7**), and considerably better than single point calculations at the semiempirical PM7-COSMO(solvent) level of theory, which provides a statistically different prediction and is considerably worse than our forcefields and XTB. Optimizations at the PM7 level were not performed as we have previously found that this leads to unphysical metallocage geometries.[23]

The accuracy of our prediction is overall modest, ~60 %, in particular with M_4L_6 cages aqueous solution, even at the tight binding DFT level of theory. Here the validity of a number of approximations needs to be considered. Among them is the assumption that both host and guest are soluble under experimental conditions. Additionally, it is assumed that conformational flexibility of the guest molecule is accurately sampled in the optimisation process. Significant errors are also expected to arise from entropic and thermal terms, which in principle should lead to an increase in accuracy but are computationally more demanding.

Finally, solvation effects, which are currently only included implicitly through the GBSA method when XTB calculations are performed, are expected to play an important role, especially in aquous or other protic solvents. Taking into account these approximations, the confidence in our prediction was estimated using an inverted gaussian function [$(1 - \exp(-(\Delta E/5k)^2)$, k = 1 kcal mol^{-1}] which value between 0 and 1. In general, true positives (TP) in the dataset have a high confidence, and consistently true negative (TN) predictions have a higher confidence than false negatives (FN), such that true binders are unlikely to be missed (**Figure 7b**). Therefore, these results provide a promising starting point from which further binding studies can be performed.

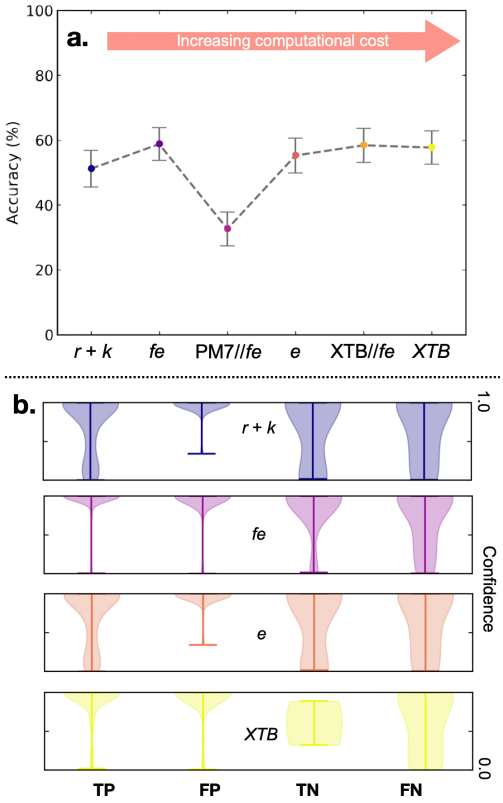


Figure 7. **(a)** Accuracy (%) in predicting a substrate binds or not considering 102 host-guest complexes reported in literature. Experimental data tabulated in **Table S2**. All binding calculations were performed within the *cgbind* module interfaced through *autodE* to MOPAC and GFN-XTB. Error bars are plotted as a combined standard deviation, considering a small statistical error (~2 kcal mol^{-1}, **Figure S6**) with an assumed implicit normal error (σ = 2 kcal mol^{-1}) and bootstrapping with replacement (10,000 resamples) on the dataset. **(b)** Confidence for true positives (TP), false positives (FP), true negatives (TN) and false negatives (FN) calculated as: $C = 1 - \exp(-(\Delta E/5k)^2)$ where k = 1 kcal mol^{-1}.

2.3.3 Combinatorial Metallocage Library

To demonstrate the ability of *cgbind* to construct a library of cages from which subsequent property, screening and filtering calculations can be performed a library of Pd_2L_4 metallocages were generated (available for download as part of the Supporting Information). Employing linkers comprised of three ends (E), 12 links (Z) and 29 centres (C) building blocks, join linearly in a E-Z-C-Z'-E' pattern and Pd^{II} as the metal centre, a theoretical total of 13,572 homoleptic cages are possible (building blocks tabulated in **Table S3**). From these, using linkers initialized with 200 conformers, 5,639 metallocages were constructed with 'reasonable' geometries (defined to be the minimum pairwise distance above 0.8 Å).[1] Considering binding of benzoquinone as a guest, cages with cavity radii smaller the centroid–O distance were removed to afford 1,353. This dataset was further reduced by eliminating those cages that were found to be much too accommodate this substrate (cavity volume >2× substrate), leading to 914 cages. Employing the algorithm electrostactic_fast to determine the optimal binding mode, 489 cages were removed based on $E_{int} > -15$ kcal mol^{-1}. Cages with formation energy [$\Delta E_f = E_C - (2E_{Pd} + 4E_L)$] above 50 kcal mol^{-1} of an optimised reference Pd_2L_4 cage reported in ref. 57 calculated using XTB single point calculations were removed affording 275 cages. The most strongly binding hosts ($\Delta E_{int} < -12$ kcal mol^{-1}) were then taken and the three most synthetically accessible determined using the method from ref. 61 assuming the most accessible linker molecules lead to the most accessible metallocages (**Figure 8b**). From these core structures monovalent atom functionalization e.g. H → Me, Et, Ph, CF_3 etc. may be performed using our Python module for molecular functionalization (github.com/duartegroup/molfunc) to afford a vast library of heteroleptic cages.

[1] A higher proportion of cages would be constructed in a reasonable geometry were the linkers initialized with more conformers

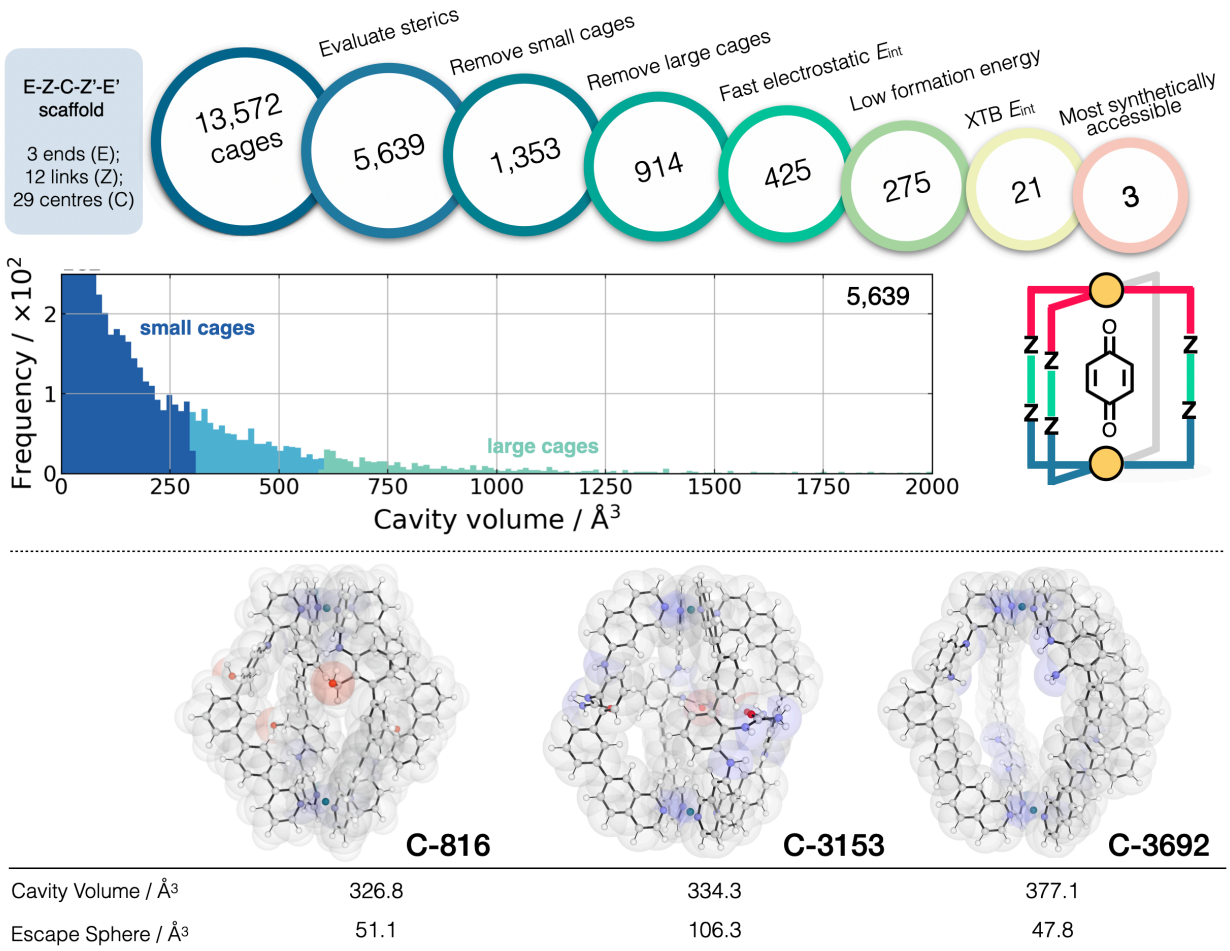


Figure 8. Filtering process used to find the three optimal hosts for benzoquinone based on simple geometric criteria, tight-binding DFT calculations and synthetic accessibility of the constituent linker molecules.

3. Conclusion

Here we have described a modular and extensible Python module *cgbind* for the automated construction and analysis of metallocage structure with arbitrary architecture (M_xL_y). Metallocages are constructed using a templating strategy and host-guest complexes generated by minimizing the energy of a simple forcefield developed here. Methods are available to analyze the resulting structures both qualitatively (electrostatic potential) and quantitively (e.g. cavity size, window size) to analyze the resulting structures. We anticipate cgbind to see utility in high-throughput metallocage screens and, through the web-app (cgbind.chem.ox.ac.uk), facilitate the rational design of metallocage with functional properties. Further work towards the reverse design of metallocages is ongoing, where rather than modifying linkers to subsequently build metallocages and search for tight complexation the cage is built around the substrate to maximize binding.

Computational Methods.

Root mean squared deviations (RMSD) were calculated using the Kabsch algorithm implemented in the *rmsd* Python module,[62] reordering atoms and not considering hydrogens, all other parameters were set to their defaults. Metallocages had to be approximately aligned manually by fitting metal atoms prior to RMSD calculation, *i.e.* the reordering of atoms was not completely effective. All *cgbind* computations were performed with v. 1.0.0 beta. Tight binding (TB-)DFT calculations were performed using GFN2-XTB v. 6.2,[41] PM7 using MOPAC2016,[42] and resolution of identify DFT using ORCA v. 4.2.[43] the latter using the PBE functional,[63] in combination with the D3BJ dispersion correction,[64,65] def2-SVP basis set[66] (which employs effective core potentials for all metals used here)[67] and the default auxiliary basis sets.[68]

Supporting Information. Information on the empirical force fields, experimental data and estimated interaction energies for the 102 complexes discussed in section 2.2.3. Root mean squared displacement values reported in Figure 6 (PDF). Coordinates of geometries obtained from *cgbind* for the $[Pd^{II}_2L_4]^{4+}$ benzoquinone complex using the simple repulsive (r+k) and electrostatic forcefields (*fe* and *e*) (Figure 5**)** (ZIP). The full library of 5,639 cages is also provided (ZIP)

Corresponding Author: FD: fernanda.duartegonzalez@chem.ox.ac.uk

Author Contributions: TY and RG wrote the code. TY and FD performed the calculations and wrote the manuscript.

Acknowledgments: We acknowledge the EPSRC Centre for Doctoral Theory and Modelling in Chemical Sciences (EP/L015722/1) for a studentship to TAY generously supported by AWE and for access to the Dirac cluster at Oxford. F.D. thank the Carnegie Trust (RIG007448) and the John Fell Oxford University Press Research Fund for financial support. We thank Alistair Sterling and Vicente Martí-Centelles for helpful discussions. This work used the Cirrus UK National Tier-2 HPC Service at EPCC (http://www.cirrus.ac.uk) funded by the University of Edinburgh and EPSRC (EP/P020267/1).

References

1 C. García-Simón, M. Costas and X. Ribas, *Chem. Soc. Rev.*, 2016, **45**, 40–62.

2 L. Chen, Q. Chen, M. Wu, F. Jiang and M. Hong, *Acc. Chem. Res.*, 2015, **48**, 201–210.

3 A. Pöthig and A. Casini, *Theranostics*, 2019, **9**, 3150–3169.

4 B. P. Burke, W. Grantham, M. J. Burke, G. S. Nichol, D. Roberts, I. Renard, R. Hargreaves, C. Cawthorne, S. J. Archibald and P. J. Lusby, *J. Am. Chem. Soc.*, 2018, **140**, 16877–16881.

5 B. Therrien, G. Süss-Fink, P. Govindaswamy, A. K. Renfrew and P. J. Dyson, *Angew. Chemie*, 2008, **120**, 3833–3836.

6 O. Zava, J. Mattsson, B. Therrien and P. J. Dyson, *Chem. - A Eur. J.*, 2010, **16**, 1428–1431.

7 J. E. M. Lewis, E. L. Gavey, S. A. Cameron and J. D. Crowley, *Chem. Sci.*, 2012, **3**, 778–784.

8 Y. Fang, J. A. Powell, E. Li, Q. Wang, Z. Perry, A. Kirchon, X. Yang, Z. Xiao, C. Zhu, L. Zhang, F. Huang and H.-C. Zhou, *Chem. Soc. Rev.*, 2019, **48**, 4707–4730.

9 C. Desmarets, T. Ducarre, M. Rager, G. Gontard and H. Amouri, *Materials (Basel).*, 2014, **7**, 287–301.

10 R. Chakrabarty, P. S. Mukherjee and P. J. Stang, *Chem. Rev.*, 2011, **111**, 6810–6918.

11 J. L. Segura, S. Royuela and M. Mar Ramos, *Chem. Soc. Rev.*, 2019, **48**, 3903–3945.

12 H. C. Erythropel, J. B. Zimmerman, T. M. de Winter, L. Petitjean, F. Melnikov, C. H. Lam, A. W. Lounsbury, K. E. Mellor, N. Z. Janković, Q. Tu, L. N. Pincus, M. M. Falinski, W. Shi, P. Coish, D. L. Plata and P. T. Anastas, *Green Chem.*, 2018, **20**, 1929–1961.

13 C. Poree and F. Schoenebeck, *Acc. Chem. Res.*, 2017, **50**, 605–608.

14 F. Neese, M. Atanasov, G. Bistoni, D. Maganas and S. Ye, *J. Am. Chem. Soc.*, 2019, **141**, 2814–2824.

15 J. Xu and A. Hagler, *Molecules*, 2002, **7**, 566–600.

16 D. J. Durand and N. Fey, *Chem. Rev.*, 2019, **119**, 6561–6594.

17 R. L. Greenaway, V. Santolini, M. J. Bennison, B. M. Alston, C. J. Pugh, M. A. Little, M. Miklitz, E. G. B. Eden-Rump, R. Clowes, A. Shakil, H. J. Cuthbertson, H. Armstrong, M. E. Briggs, K. E. Jelfs and A. I. Cooper, *Nat. Commun.*, 2018, **9**, 2849.

18 J. G. Freeze, H. R. Kelly and V. S. Batista, *Chem. Rev.*, 2019, **119**, 6595–6612.

19 M. Foscato and V. R. Jensen, *ACS Catal.*, 2020, **10**, 2354–2377.

20 A. R. Rosales, J. Wahlers, E. Limé, R. E. Meadows, K. W. Leslie, R. Savin, F. Bell, E. Hansen, P. Helquist, R. H. Munday, O. Wiest and P.-O. Norrby, *Nat. Catal.*, 2019, **2**, 41–45.

21 Y. Guan, V. M. Ingman, B. J. Rooks and S. E. Wheeler, *J. Chem. Theory Comput.*, 2018, **14**, 5249–5261.

22 J. Hachmann, R. Olivares-Amaya, S. Atahan-Evrenk, C. Amador-Bedolla, R. S. Sánchez-Carrera, A. Gold-Parker, L. Vogt, A. M. Brockway and A. Aspuru-Guzik, *J. Phys. Chem. Lett.*, 2011, **2**, 2241–2251.

23 T. A. Young, V. Martí-Centelles, J. Wang, P. J. Lusby and F. Duarte, *J. Am. Chem. Soc.*, 2020, **142**, 1300–1310.

24 C. E. Wilmer, M. Leaf, C. Y. Lee, O. K. Farha, B. G. Hauser, J. T. Hupp and R. Q. Snurr, *Nat. Chem.*, 2012, **4**, 83–89.

25 C. M. Simon, R. Mercado, S. K. Schnell, B. Smit and M. Haranczyk, *Chem. Mater.*, 2015, **27**, 4459–4475.

26 S. P. Ong, W. D. Richards, A. Jain, G. Hautier, M. Kocher, S. Cholia, D. Gunter, V. L. Chevrier, K. A. Persson and G. Ceder, *Comput. Mater. Sci.*, 2013, **68**, 314–319.

27 T. F. Willems, C. H. Rycroft, M. Kazi, J. C. Meza and M. Haranczyk, *Microporous Mesoporous Mater.*, 2012, **149**, 134–141.

28 B. J. Bucior, A. S. Rosen, M. Haranczyk, Z. Yao, M. E. Ziebel, O. K. Farha, J. T. Hupp, J. I. Siepmann, A. Aspuru-Guzik and R. Q. Snurr, *Cryst. Growth Des.*, 2019, **19**, 6682–6697.

29 A. S. Rosen, J. M. Notestein and R. Q. Snurr, *J. Comput. Chem.*, 2019, **40**, 1305–1318.

30 M. Miklitz, S. Jiang, R. Clowes, M. E. Briggs, A. I. Cooper and K. E. Jelfs, *J. Phys. Chem. C*, 2017, **121**, 15211–15222.

31 L. Turcani, E. Berardo and K. E. Jelfs, *J. Comput. Chem.*, 2018, **39**, 1931–1942.

32 A. Sturluson, M. T. Huynh, A. H. P. York and C. M. Simon, *ACS Cent. Sci.*, 2018, **4**, 1663–1676.

33 O. Kravchenko, A. Varava, F. T. Pokorny, D. Devaurs, L. E. Kavraki and D. Kragic, *J. Chem. Inf. Model.*, 2020, **60**, 1302–1316.

34 M. Miklitz and K. E. Jelfs, *J. Chem. Inf. Model.*, 2018, **58**, 2387–2391.

35 S. van der Walt, S. C. Colbert and G. Varoquaux, *Comput. Sci. Eng.*, 2011, **13**, 22–30.

36 P. Virtanen, R. Gommers, T. E. Oliphant, M. Haberland, T. Reddy, D. Cournapeau, E. Burovski, P. Peterson, W. Weckesser, J. Bright, S. J. van der Walt, M. Brett, J. Wilson, K. J. Millman, N. Mayorov, A. R. J. Nelson, E. Jones, R. Kern, E. Larson, C. Carey, İ. Polat, Y. Feng, E. W. Moore, J. VanderPlas, D. Laxalde, J. Perktold, R. Cimrman, I. Henriksen, E. A. Quintero, C. R. Harris, A. M. Archibald, A. H. Ribeiro, F. Pedregosa, P. van Mulbregt and S. 1. 0 Contributors, *arXiv*, 2019, 1–22.

37 A. Hagberg, S. D and P. Swart, in *Proceedings of the 7th Python in Science conference*, 2008, pp. 11–15.

38 D. Weininger, *J. Chem. Inf. Model.*, 1988, **28**, 31–36.

39 G. Landrum, *RDKit: Open-source cheminformatics, v. 2019*, GitHub (https://github.com/rdkit/rdkit), 2019.

40 S. Riniker and G. A. Landrum, *J. Chem. Inf. Model.*, 2015, **55**, 2562–2574.

41 C. Bannwarth, S. Ehlert and S. Grimme, *J. Chem. Theory Comput.*, 2019, **15**, 1652–1671.

42 J. J. P. Stewart, *MOPAC2016*, Stewart Computational Chemistry, Colorado Springs, CO, USA, 2016.

43 F. Neese, *WIREs Comput. Mol. Sci.*, 2018, **8**, 1–6.

44 M. Valiev, E. J. Bylaska, N. Govind, K. Kowalski, T. P. Straatsma, H. J. J. Van Dam, D. Wang, J. Nieplocha, E. Apra, T. L. Windus and W. A. de Jong, *Comput. Phys. Commun.*, 2010, **181**, 1477–1489.

45 V. Martí-Centelles, A. L. Lawrence and P. J. Lusby, *J. Am. Chem. Soc.*, 2018, **140**, 2862–2868.

46 W. Kabsch, *Acta Crystallogr. Sect. A*, 1976, **32**, 922–923.

47 D. L. Caulder, R. E. Powers, T. N. Parac and K. N. Raymond, *Angew. Chemie Int. Ed.*, 1998, **37**, 1840–1843.

48 R. Zhu, W. M. Bloch, J. J. Holstein, S. Mandal, L. V. Schäfer and G. H. Clever, *Chem. - A Eur. J.*, 2018, **24**, 12976–12982.

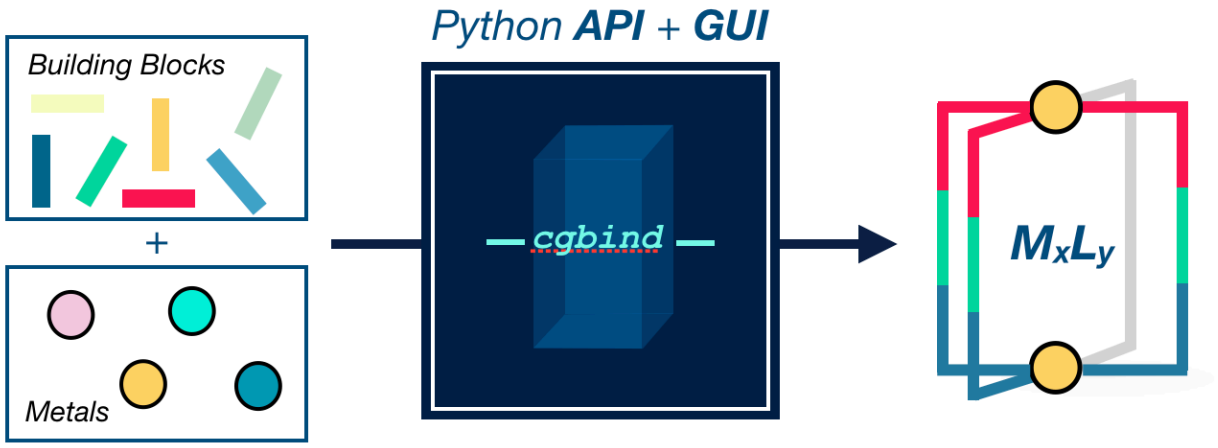
49 D. C. Young, *Computational Chemistry*, John Wiley & Sons, Inc., New York, USA, 2001.

50 D. J. Wales and J. P. K. Doye, *J. Phys. Chem. A*, 1997, **101**, 5111–5116.

51 S. Grimme, C. Bannwarth and P. Shushkov, *J. Chem. Theory Comput.*, 2017, **13**, 1989–2009.

52 H.-X. Zhou and M. K. Gilson, *Chem. Rev.*, 2009, **109**, 4092–4107.
53 C. Jiang, X. Jin, Y. Dong and M. Chen, *J. Chem. Inf. Model.*, 2016, **56**, 1132–1138.
54 N. Rego and D. Koes, *Bioinformatics*, 2015, **31**, 1322–1324.
55 N. M. O'Boyle, T. Vandermeersch, C. J. Flynn, A. R. Maguire and G. R. Hutchison, *J. Cheminform.*, 2011, **3**, 8.
56 J.-P. Ebejer, G. M. Morris and C. M. Deane, *J. Chem. Inf. Model.*, 2012, **52**, 1146–1158.
57 D. P. August, G. S. Nichol and P. J. Lusby, *Angew. Chemie Int. Ed.*, 2016, **55**, 15022–15026.
58 R. L. Spicer, A. D. Stergiou, T. A. Young, F. Duarte, M. D. Symes and P. J. Lusby, *J. Am. Chem. Soc.*, 2020, **142**, 2134–2139.
59 T. K. Ronson, W. Meng and J. R. Nitschke, *J. Am. Chem. Soc.*, 2017, **139**, 9698–9707.
60 M. M. J. Smulders, S. Zarra and J. R. Nitschke, *J. Am. Chem. Soc.*, 2013, **135**, 7039–7046.
61 P. Ertl and A. Schuffenhauer, *J. Cheminform.*, 2009, **1**, 8.
62 J. C. Kromann, *Calculate Root-mean-square deviation (RMSD) of Two Molecules Using Rotation. v.1.3.0*, GitHub (https://github.com/charnley/rmsd), 2019.
63 J. P. Perdew, K. Burke and M. Ernzerhof, *Phys. Rev. Lett.*, 1996, **77**, 3865–3868.
64 S. Grimme, J. Antony, S. Ehrlich and H. Krieg, *J. Chem. Phys.*, 2010, **132**, 154104.
65 S. Grimme, S. Ehrlich and L. Goerigk, *J. Comput. Chem.*, 2011, **32**, 1456–1465.
66 F. Weigend and R. Ahlrichs, *Phys. Chem. Chem. Phys.*, 2005, **7**, 3297.
67 D. Andrae, U. Häußermann, M. Dolg, H. Stoll and H. Preuß, *Theor. Chim. Acta*, 1990, **77**, 123–141.
68 F. Weigend, *Phys. Chem. Chem. Phys.*, 2006, **8**, 1057.

Table of Contents



Accurate Geometries • Automated Calculations • Host-Guest Complex